```
    // if the leap-years are in between
    // the YEAR of birth and the actual Year.
    // c++ is counting the leap-years
    // within this two arguments.
    // The third IF-statement is checking if
    // the actual MONTH is greater than 2 AND
    // the actual YEAR is a leap year.
    // If this is true "d" is 1 otherwise zero.

    for (h=1900; h <getYear()+1900 ; h=h+4) {
       if ((h == GJahrOut)&(GMonatOut 3)&(GTagOut 29)) {
        b = 1;
      }
      if ((h  GJahrOut)&(h now.getYear()+1900)) {
        c++;
      }
       if ((h == now.getYear()+1900)&(now.getMonth()+1  2)) {
        d = 1;
      }
    }

    // STage      (Integer)    value of leap-year days

    int STage = (b+c+d);

    // LTage   (Double)  value of life-days
    // TG    (Double)  draw LTage in the correct scale
    // VS    (Integer) draw the y-axe, normally the left border
    //       of the applet, shifted to the middle of viewing area.

    double LTage= (STage+JTage);
    double TG=LTage*masstab;
    int VS=12*masstab;

    // additional information, !! not necessarily for the program !!

    g.drawString("actual date is the system date !!! ", sx, sy);
    g.drawString("You are " + (LTage) + " days old ", sx, sy+20);

    g.setColor(Color.red);
    g.drawString("RED represents the emotional fitness", sx+300,
sy);
    g.setColor(Color.green);
    g.drawString("GREEN represents the vital fitness", sx+300,
sy+14);
    g.setColor(Color.blue);
    g.drawString("BLUE represents the mental fitness", sx+300,
sy+28);

    // drawing the function of the three biorythm-curves
    // emotional rythm has a period of 28 days and the function [f0]

    // vital rythm has a period of 23 days and the function [f1]
    // mental rythm has a period of 33 days and the function [f2]

    for (int x = 0 ; x size().width ; x++) {
      g.setColor(Color.red);
      g.drawLine(x, (int)f0(x+TG-VS), x + 1, (int)f0(x + 1 +
TG-VS));
      g.setColor(Color.green);
      g.drawLine(x, (int)f1(x+TG-VS), x + 1, (int)f1(x + 1 +
TG-VS));
      g.setColor(Color.blue);
      g.drawLine(x, (int)f2(x+TG-VS), x + 1, (int)f2(x + 1 +
TG-VS));
    }

    // This statement sets the color back to black
    g.setColor(Color.black);

    // draw vertical line as dots, each dot representing one day

    for (int a = 0 ; a size().width ; a=a+masstab) {
      g.drawLine(a, (2*size().height/4), a,(2*size().height/4));
     }

    // draw vertical lines in distances of 2 weeks,
    // begining at actual date
     //TimeLine      (Integer)    is increment of the loop

    int TimeLine = 14*masstab;

      for (int r = VS ; r size().width ; r=r+TimeLine) {
      g.drawLine(r, (2*size().height/4), r,(3*size().height/4));
    }

    // Z  (Integer) useful to center string to vertical line

    int Z=2*masstab;

    g.drawString (now.getDate()+"."+(now.getMonth()+1)+"."+
                  (now.getYear()+1900),sx+VS-Z,sy+200);
    g.drawString("next 2 weeks",sx+VS+TimeLine-Z,sy+200);
    g.drawString("next 4 weeks",sx+VS+2*TimeLine-Z,sy+200);
  }

  public void redraw(boolean filled, int aa, int ab, int ac) {

    this.GJahrOut = aa;
    this.GMonatOut = ab;
    this.GTagOut = ac;
    repaint();
  }
}

class EingabeGebDaten extends Panel {

  //  List of variables
  //  GJahr,GMonat,GTag (TextField)  draws input field for birth
data
  //  Ausgabe       (Canvas)   creates a canvas

  TextField GJahr, GMonat, GTag;
  Ausgabe canvas;

  public EingabeGebDaten(Ausgabe canvas) {
    this.canvas = canvas;
    add(GJahr = new TextField("Year", 4));
    add(GMonat = new TextField("Month", 4));
    add(GTag = new TextField("Day", 4));
    add(new Button("Confirm Input"));
  }

  public boolean action(Event ev, Object arg) {
    if (ev.target instanceof Button) {
    String label = (String)arg;
     canvas.redraw(label.equals("Confirm Input"),
        Integer.parseInt(GJahr.getText().trim()),
        Integer.parseInt(GMonat.getText().trim()),
        Integer.parseInt(GTag.getText().trim()));
    return true;
    }
   return false;
  }
}
```

## Spezialtasten

| | |
|---|---|
| **Alt** | Diese Taste muss ersetzt werden |
| **AltGr** | Diese Taste muss SOFORT ersetzt werden |
| **Backspace** | Arschspalte |
| **Break** | Tastatur muss erbrechen |
| **CapsLock** | Ausgehverbot fuer Kapitaene |
| **Ctrl** | Taste fuer Konsonanten |
| **Enter** | maennliche Ente |
| **Home** | nach Hause gehen |
| **PageDown** | Vorlage ist heruntergefallen |
| **PageUp** | Wieder einlesen |
| **Pause** | Kaffee ist fertig |
| **Shift** | es regnet |